Utility Patent Application

of

Steven B. BOWLER

for

CROSS-PROGRAM DEPENDENCY SCHEDULING

Copyright Notice

Cross Reference to Related Applications

[0002] Applicant claims priority of U.S. Provisional Patent Application in the name of Steven B. Bowler, application number 60/421,025, filed on 10-24-2002.

Background of the Invention

[0003] Field of the Invention

[0004] This invention relates to new product development, and particularly to Internet software that enables collaborative program management during new product development.

[0005] Description of Prior Art

[0006] Program Management includes scheduling of activities and their interdependencies, raising and resolving issues, managing the creation and release of critical intellectual property, building prototypes, and releasing products to full production. The overall product life cycle is managed in this manner from conception to conclusion. Program management resources include metrics, problem logs, alerts, team meetings, phase exit reviews, and audits. These resources are used to keep a large-scale program on track, to put the program on hold, or to kill the program altogether to minimize loss.

[0007] History of Collaborative Scheduling Programs.

[0008] In the past, scheduling of a program or project involved the creation of a timing chart, called a Gantt diagram, where a hierarchy of activities is displayed graphically along an axis of time. The Gantt diagram assisted program managers and team members to visualize the interdependencies of their activities. The diagram displayed task names, resources required for such tasks, and dates for starting and ending various activities.

[0009] As technologies evolved, computer applications for scheduling emerged such as FastTrack.RTM Schedule for Macintosh.RTM and Microsoft.RTM Project.RTM for the personal computer. These applications produced the required graphical output of the Gantt diagrams, but they suffered from a poor ability to communicate the schedule data to program managers and team members. Schedule data was accessible from only one computer, and the only output option was a printer.

[0010] Accessibility of schedule data improved with client-server architectures and applications. Companies such as Artemis and Primavera developed applications that enabled sharing of schedules among licensed users in the same IT (Information

Technology) infrastructure. Using applications to share schedules outside company boundaries and IT infrastructures usually resulted in miscommunication or even no communication.

[0011] A new breed of applications using operating system-independent power of the Internet began to emerge. These applications used enterprise-class relational databases that could link program data in newer relational ways. At the same time, professional service organizations were advocating the use of technology bookshelf processes to separate the development of pure technology from the delivery of new product designs to speed new products to market. A new technology, once tested and approved, could then feed multiple platform products. This meant that multiple programs could be dependent on the completion of the same predecessor technology development. Traditional Project Management only handled dependencies inside the project plan. Thus, critical developments outside the program could adversely affect profitability without warning.

[0012] What is needed, therefore, is a scheduling system capable of creating, displaying, and managing cross-program dependencies. What is further needed is such a scheduling system that communicates with a database and has a graphical user interface to display how schedule and status of an activity in one program can impact the schedule and status of an activity in another program. What is also need is such a scheduling system that simultaneously implements fixed-duration scheduling with electronic notification.

Brief Summary of the Invention

[0013] This invention pertains to software for the mechanics of scheduling. This involves scheduling algorithms that cross traditional program boundaries, providing a new perspective on scheduling, which is especially useful for enterprises managing multiple dependent programs.

[0014] The present invention exists as a stand-alone application and can also be a component of a broader set of software modules that comprise an overall program management suite. The present invention is directed to a scheduling system that establishes cross-program dependencies for phase, task, deliverable, and gate activities. The scheduling system includes Internet software that displays Gantt charts, organizes online meetings, sends e-mail alerts, and provides document storage and retrieval. Dependencies and status of activities are displayed in real-time for access by all program managers.

[0015] Modification of an activity by one manager causes the system to notify all managers affected by the modification. Alternatively, the system uses permission constraints to prevent one program manager from modifying or negatively impacting another's program without a closed-loop notification and approval. Before a schedule can change in one program that adversely affects the schedule of another program due to fixed schedule durations and finish-to-start dependencies, a notification is sent to the affected program manager. Program managers in multiple programs can be immediately alerted to and respond appropriately to slippage in dependent programs outside of their control. If a program manager accepts the schedule slip, then the program is automatically rescheduled. If the program manager rejects the schedule slip, then the conflict is identified in a message back to the originating program manager, and the two must resolve the schedule conflict. Each program activity will carry a section that shows these cross-program dependencies and flags

for schedule conflicts. The alerts will roll up to the highest-level owner where the schedule conflict has an impact.

[0016] Objects

[0017] It is an object of the present invention to create a schedule system that enables collaborative program management during product development.

[0018] It is another object of the present invention to create such a scheduling system that communicates dependencies in any program in the same database.

[0019] Another object of the present invention is to create such a scheduling system that prevents one program manager from negatively impacting another's program without a closed-loop notification.

[0020] Features

[0021] A feature of the present invention is a rescheduling algorithm that allows permission constraints.

[0022] Another feature of the present invention is a graphical display of program status information and ownership information of dependent or related programs.

[0023] Advantages

[0024] There are many advantages to the invention, including the following. Other advantages will be apparent to those skilled in the art.

[0025] An advantage of the present invention is real time interaction between dependent schedules that were previously managed separately.

[0026] Another advantage is that program managers and senior executives can make informed decisions and execute their decisions instantaneously. All parties affected by schedule conflicts receive instant notification.

[0027] Additional advantages include optimized allocation of resources, simple establishment of dependencies, single IT platform, and automated schedule conflict resolution.

Brief Description of the Drawings

[0028] **FIG.** 1 is a flow chart of the history of the development of collaborative scheduling programs.

[0029] **FIG. 2** is a flow chart of the decision-making process involving cross program dependencies to alert manager of slips in one program that impact another program.

[0030] **FIG. 3** Is a status display of the scheduling system showing an outside task not impacting on the critical path of the focus program.

[0031] **FIG. 4** Is a status display of the scheduling system showing an outside task impacting the critical path of the focus program.

[0032] **FIG. 5** Is a status display of the scheduling system showing tasks from the focus program automatically rescheduled from the slip of an outside task.

Detailed Description of the Invention, Including the Preferred Embodiment

[0033] In the following detailed description of the invention, reference is made to the accompanying drawings, which form a part hereof, and in which are shown, by way of illustration, specific embodiments in which the invention may be practiced. It is to be understood that other embodiments may be utilized, and structural changes may be made without departing from the scope of the present invention.

[0034] The invention is implemented in a software application written in any number of programing languages such as Java, VisualBasic, or C++. The invention is preferably written in Java because Java ports well from one operating system to another, which means that Java applications can run on any computer that has implemented the Java virtual machine. The preferred platform is one capable of running a web server (such as Apache, Microsoft IIS, or Weblogic) and a database server, such as Microsoft Windows2000 operating system running in a personal computer. Other possible platforms include Unix-based platforms such as those sold by Sun Microsystems, Linux-based systems, or MacOS-based systems.

[0035] In **FIG. 2**, Program manager A and Program manager B create separate programs with several tasks. Program manager A determines task A3 cannot begin until completion of task B4. Program manager A establishes a cross-program dependency between task A3 and task B4. As task B4 slips past its due date, Program manager A automatically receives an e-mail alert, and Program manager A can choose to reschedule or add resources. Program manager A elects to reschedule task A3.

[0036] Referring now to **FIG. 3** through **FIG. 5**, the scheduling system is shown according to the preferred embodiment. The scheduling system is shown in Gantt chart format to show a sequence of events. In **FIG. 3**, the dependent task B4 from program B is

shown on the Gantt chart for Program A. Task B4 is not impacting on the Critical Path of Program A, so no action is required. In **FIG. 4**, a slip in schedule for Task B4 in Program B is now impacting on Task A3 in Program A. In the **FIG. 5**, the Program manager for Program A has rescheduled Program A to recognize the slip caused in Program B.